WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, February 06, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L10	L9 and (message adj3 (number or count))	10
☐	L9	L8 and (packet adj2 length)	110
☐	L8	L7 and (packet adj2 (number or count or counter))	191
☐	L7	L5 and (checksum or CRC)	442
☐	L6	L5 and checksum	196
☐	L5	19990318	1297
☐	L4	packet near8 ((to or destination) adj2 address) near8 ((from or source) adj2 address)	2967
☐	L3	packet near8 ((to or destination) adj2 address) near8 ((from or source) adj2 address) near8 (command or function) near8 (data or information) near8 checksum near8 (packet adj2 (number or count or counter)) near8 length near8 (message adj3 (number or count))	0
		DB=USPT; PLUR=YES; OP=ADJ	
☐	L2	L1 and (request same host)	1
☐	L1	6124806.pn.	1

END OF SEARCH HISTORY

First Hit Fwd Refs

Generate Collection Print

L10: Entry 4 of 10 File: USPT Sep 30, 1997

DOCUMENT-IDENTIFIER: US 5673252 A
**** See image for Certificate of Correction ****
TITLE: Communications protocol for remote data generating stations

Application Filing Date (1):
19950526

Brief Summary Text (34):
When a first remote cell node is polled with a first polling signal by the intermediate data terminal, neighboring remote cell nodes receive the RCN-packet signal transmitted by the first remote cell node. Upon receiving an acknowledgment signal from the intermediate data terminal, at the neighboring remote cell nodes, the respective RCN-processor means deletes from the respective RCN-memory means messages, i.e., NSM-packet signals, received from the network service modules that have the same message identification number as messages transmitted in the RCN-packet signal from the first remote cell node to the intermediate data terminal.

Brief Summary Text (37):
The IDT-memory means stores the received RCN-packet signals. The IDT-processor means collates the NSM-packet signals embedded in the RCN-packet signals received from the multiplicity of remote cell nodes, identifies duplicates of NSM-packet signals and deletes the duplicate NSM-packet signals, i.e., messages from network service modules that have the same message identification number. In response to a second polling signal from a central data terminal, the IDT-transmitter means transmits the stored multiplicity of received RCN-packet signals as an IDT-packet signal to the central data terminal.

Detailed Description Text (12):
The NSM-transmitter means is embodied as an NSM transmitter 318. The NSM transmitter 318 transmits at a first carrier frequency, using radio waves, the respective NSM data from the physical device in brief message packets called NSM-packet signals. The NSM-packet signal might have a time duration of 100 milliseconds, although other time durations can be used to meet particular system requirements. The NSM-packet signal transmitted by the NSM transmitter 318 follows a generic or fixed format, and a representative message packet is illustrated in FIG. 3. Included in the message is: preamble; opening frame; message type; message identification; service module type; message number; service module address; data field; error detection; and closing frame.

Detailed Description Text (74):
When a first remote cell node is polled with a first polling signal by the intermediate data terminal, neighboring remote cell nodes 112 receive the RCN-packet signal transmitted by the first remote cell node. Upon receiving an acknowledgment signal from the intermediate data terminal that polled the first remote cell node, at the neighboring remote cell nodes 112 the respective RCN processor deletes from the respective RCN memory messages from the network service modules that have the same message identification number as messages transmitted in the RCN-packet signal from the first remote cell node to the intermediate data terminal. The message identification number is illustrated in a typical NSM-data packet in FIG. 3.

Detailed Description Text (82):
(e) A message number so that the messages are numbered sequentially. In this way, again, the remote cell node 112 can determine whether a message has been lost or whether the information received is merely a duplicate message from a duplicate one of the receiving stations.

Detailed Description Text (101):
The IDT memory 515 stores the received RCN-packet signals. The IDT processor 514 collates the NSM-packet signals embedded in the RCN-packet signals received from the multiplicity of remote cell nodes, identifies duplicates of NSM-packet signals and deletes the duplicate NSM-packet signals, i.e., messages from network service modules that have the same message identification number.

Detailed Description Text (186):
The physical layer provides data transfer and time reference services to higher layers. The physical layer at the network service module provides a number of services to the data link and network layers. These services include: obtaining network synchronization, as explicitly requested by a higher layer; maintaining current time, between synchronizations, and returning it upon request; checking for and, if present, receiving a packet of specified length, or receiving bytes until told otherwise, from a specified channel or subchannel; locating the start of a specified slot within a specified channel and transmitting a specified preamble followed by a series of bytes of a data link packet passed to it; and detecting abnormal conditions and aborting its action with an appropriate status return code upon detection of such abnormal conditions.

Detailed Description Text (202):
flags for today and tomorrow, indicating whether day light savings time is in effect; and CRC, 16 bits.

Detailed Description Text (203):
Every 30 seconds, intermediate data terminals perform an internal synchronization procedure which, since the synchronization procedure involves the use of the intermediate data terminals' RF receiver and transmitter, can be performed during RND/NRR slots. Ideally the synchronization procedure should occur just before the IRS slot in the S channel and, therefore the synchronization procedure is scheduled to occur during the first half of channel 28. Although time information could be delivered to remote cell nodes in some other fashion, since the frame number is needed and must also be protected, i.e., with a CRC, having at least as many data bits in the packet as there are bits in the CRC is not a drawback.

Detailed Description Text (214):
The RNS slot fields need not be CRC protected. The synchronization is fixed, and successive count fields are sequential values and are inherently redundant.

Detailed Description Text (226):
CRC, e.g., 16 bits.

Detailed Description Text (232):
The data link layer does perform data link address recognition, including global broadcasts. It also provides error control by including, in all packets, a CRC check field that is verified upon reception. Packets failing CRC verification are usually counted and discarded. Remote cell nodes also provide the option of capturing a packet as a digitized analog signal.

Detailed Description Text (234):
The intermediate data terminal--remote cell node link is a more conventional master/slave polled access link or, in HDLC parlance, an unbalanced normal response mode. As master, the intermediate date terminal is responsible for protocol

integrity; initiating all dialogues, determining when retransmission, in either direction, is necessary, and taking corrective actions when the dialogue gets out of synchronization. Data link address recognition and CRC error detection are conventional. Packets sequence numbers; acknowledgements, by returning the sequence number of the last packet successfully received; and packet retransmission are employed. Separate sequence numbers are used for individually addressed and broadcast streams of packets. Flow control is exercised inasmuch as each packet must be acknowledged before the next one can be sent, except for intermediate data terminal to remote cell node broadcasts and RIQs.

Detailed Description Text (243):
Error Detection: typically a CRC is used.

Detailed Description Text (254):
Received signals match particular physical synchronization patterns which prefix the message before being considered as packets, thus filtering out noise and some corrupted messages. Bit patterns used to create the frame/channel structure also are selected to prevent misinterpretation, either due to noise or because some part of a legitimate message looks the same. In general, a two level scheme may be employed where two patterns, separated by a fixed time interval, match. All packets are further protected by encoding them using a cyclic code, 16-bit CRC, which affords a degree of information redundancy. Although not required, an error correcting code can be used to recover from certain types of error, e.g., to achieve single error correction with a Hamming code. Single error correction could also be achieved with just the CRC, at considerable computational cost, using a brute force search method.

Detailed Description Text (262):
Each of the various wide area communications network links has a data link packet structure which is loosely modeled on the HDLC format, but optimized to the special needs and purposes of the particular link. When designing data link packets generally, a physical layer preamble precedes the packet; this preamble is required to recognize the start of a slot. The bit values of a received preamble may or may not be made available, and packets end at the final bit of the CRC. However, the physical layer requires that a transmitter transmit at least one bit having an arbitrary value after the final bit of the CRC. A flag pattern marks the start of a data link packet. This flag pattern is considered part of the data link packet, but the physical layer also uses the flag pattern to recognize the start of the slot and then makes this field available to the data link layer.

Detailed Description Text (263):
When designing data link packets of a specific type, packets except for IRD and RIR are a fixed length, and packets sizes are byte multiples, except RND and RNC. An IRD link packet is special in that it is always paired with the previous IRH packet, as shown in FIG. 26. RND link packets are special in that their structure depends on CAT subchannel assignment. Additionally, each link has associated with it an intrinsic form of addressing: NRR, RIR and RIQ packets contain source node address fields; IRH packets contain destination node address fields which may be broadcast address; IRD packets implicitly refer to the same destination address as the prior IRH packet; RND addressing depends on subchannel use designated by the CAT; and RNC is implicitly a broadcast-only link. Finally, the data link layer shares a control field with the network layer, all packets are protected by a 16-bit CRC check field, and cardinal numbers are transmitted most significant byte and, bit within byte, first. FIGS. 27-30 illustrate RIR date link packet structure, IRH data link packet structure, NRR data link packet structure, and RIQ data link packet structure, respectively.

Detailed Description Text (275):
number of packets received with valid CRCs

Detailed Description Text (276):
number of packets with CRC errors

Detailed Description Text (278):
Additionally, higher layers can ask the data link layer to relay packets with CRC errors or packets introduced with a bad preamble along with CRC validated packets, in decoded binary form, or for any packet as a digitized analog signal.

Detailed Description Text (281):
Services provided to the network layer 96 include encapsulating a network message in a packet and transmitting the network message in a particular slot; receiving packets from pre-specified sloes, verifying CRCs, extracting and buffering the network message fields, and returning them upon request; returning current time; and returning status and operational measurements, including those from the physical layer 98.

Detailed Description Text (430):
msgno--message sequence number. Increments modulo 16 with each message transmitted by a network service module, independent of msgtype. Used by the network layer to help identify and count lost messages and, along with msgtype, for eavesdropping; msgno is not used for message acknowledgement or retransmission.

Detailed Description Text (454):
Remote cell nodes respond to polls for messages by sending a block of up to five messages at a time. These RCN report messages are structured to minimize receiver energy of neighboring remote cell nodes who are using eavesdropping to perform message redundancy control. The fields from the NSM message which uniquely identify it are placed first in the RCN report, followed by an intermediate CRC. Eavesdropping remote cell nodes can stop listening once they receive this CRC. The rest of the NSM message content comes after that. Remote cell node transmit energy is further minimized by making these report messages variable length. The maximum number of NSM messages which fit in the report depends on how many additional tag fields are requested by the intermediate data terminal, and the report message size varies because an integer number of tagged NSM messages may be smaller than the maximum size of the network message field of an RIR data link packet. Remote cell nodes which only have fewer than this number of NSM messages to relay, transmit a shorter report message. FIGS. 46-49 illustrate RIR network message format used to relay NSM messages in the context of a data line packet, RIR network message subfields comprising the data link control field, subfields comprising the RCN status field, and the maximum number of NSM messages per RIR, respectively.

Detailed Description Text (461):
seqind--message sequence number (per individual RCN) (4)

Detailed Description Text (465):
intermediate CRC--from start of data link packet

Detailed Description Text (477):
etag--indicates message received with CRC error (1)

Detailed Description Text (481):
crc--original CRC received with NSM message (16)

Detailed Description Text (506):
For broadcast, due to the long delay to acquire acknowledgements, a sliding window flow control strategy is proposed, using the selective reject ARQ technique for error control. A 4-bit sequence number allows up to eight outstanding messages to be unambiguously ACKed or NAKed selectively. An ACK-n acknowledges all outstanding messages with sequence numbers less than n, up to eight, and says nothing about message n, while a NAK-n also acknowledges messages less than n but explicitly

requests a retransmission of message with sequence number n.

Detailed Description Text (538):
The RNC slot is intended for very occasional use to deliver a very limited amount of command information to network service modules with a very short delay. Network service modules are expected to be listening to every RNC slot. The remote cell node simply takes the SAC field from the IRH, surrounds it with an opening flag and CRC, and transmits.

L10: Entry 10 of 10 File: USPT May 1, 1990

DOCUMENT-IDENTIFIER: US 4922486 A
TITLE: User to network interface protocol for packet communications networks

Application Filing Date (1):
19880331

Brief Summary Text (19):
Protocols such as X.25 have been aimed primarily for use in systems in which basic data transfer messages transmit relatively small numbers of data blocks each limited in length to typically several thousand bytes for each block. As a result, these systems, in the interest of efficiency, limit the length of header information that is transmitted with each data block to that information which is most critically needed for handling data transfers.

Detailed Description Text (82):
The external link protocol between the NIM and MINT allows the XLH 16 to detect the beginning and end of network transactions. The transactions are immediately moved into a memory 18 desinged to handle the 150 Mb/s bursts of data arriving at the XLH. This memory access is via a high-speed time slotted ring 19 which guarantees each 150 Mb/s XLH input and each 150 Mb/s output from the MINT (i.e. MANS inputs) bandwidth with no contention. For example, a MINT which concentrates 4 remote NIMs and has 4 input ports to the center switch must have a burst access bandwidth of at least 1.2 Gb/s. The memory storage is used in fixed length blocks of a size equal to the maximum packet size plus the fixed length MAN header. The XLH moves an address of a fixed size memory block followed by the packet or SUWU data to the memory access ring. The data and network header are stored until the MINT control 20 causes its transmission into the MANS. The MINT control 20 will continually supply the XLHs with free memory block addresses for storing the incoming packets and SUWUs. The XLH also "knows" the length of the fixed size network header. With this information the XLH passes a copy of the network header to MINT control 20. MINT control 20 pairs the header with the block address it had given the XLH for storing the packet or SUWU. Since the header is the only internal representation of the data within MINT control it is vital that it be correct. To ensure sanity due to potential link errors the header has a cyclic redundancy check (CRC) of its own. The path this tuple takes within MINT control must be the same for all packets of any given LUWU (this allows ordering of LUWU data to be preserved). Packet and SUWU headers paired with the MINT memory block address will move through a pipeline of processors. The pipeline allows multiple CPUs to process different network transactions at various stages of MINT processing. In addition, there are multiple piplines to provide concurrent processing.

Detailed Description Text (283):
At the same time a packet is being transmitted on the ring, the header of the packet is deposited in the header FIFO 266 that is subsequently read by the XLH processor 268. In this header are the source and destination address fields, which the central control will require for routing. In addition, the header checksum is verified to ensure that these fields have not been corrupted. The header information is then packaged with a memory block descriptor (address and length)

and sent in a message to the central control 20 (FIG. 4).

Detailed Description Text (316):
The header for each packet entering an XLH is transmitted along with the address where that packet is being stored directly to an associated XLH manager 305, if the header has passed the hardware check of the cyclic redundancy code (CRC) of the header performed by the XLH. If that CRC check fails, the packet is discarded by the XLH which recycles the allocated memory block. The XLH manager passes the header and the identity of allocated memory for the packet to the source checker 307. The XLH manager recycles memory blocks if any of the source checker, router, or NIM queue manager find it impossible to transmit the packet to a destination. Recycled memory blocks get used before memory blocks allocated by the memory manager. Source checker 307 checks whether the source of the packet is properly logged in and whether that source has access to the virtual network of the packet. Source checker 307 passes information about the packet, including the packet address in MINT memory, to router 309 which translates the packet group identification, effectively a virtual network name, and the destination name of the packet in order to find out which output link this packet should be sent on. Router 309 passes the identification of the output link to NIM queue manager 311 which identifies and chains packets received by the four XLHs of this MINT which are headed for a common output link. After the first packet to a NIM queue has been received, the NIM queue manager 311 sends a switch setup request to switch setup control 313 to request a connection to that NIM. NIM queue manager 311 chains these packets in FIFO queues 316 of switch unit queue 314 so that when a switch connection is made in the circuit switch 10, all of these packets may be sent over that connection at one time. Output control signal distributor 138 of the switch control 22 replies with an acknowledgement when it has set up a connection. This acknowledgment is received by switch setup control 313 which informs NIM queue manager 311. NIM queue manager 311 then informs ILH 17 of the list of chained packets in order that ILH 17 may transmit all of these packets. When ILH 17 has completed the transmission of this set of chained packets over the circuit switch, it informs switch setup control 313 to request a disconnect of the connection in switch 10, and informs memory manager 301 that the memory which was used for storing the data of the message is now available for use for a new message. Memory manager 301 sends this release information to memory distributor 303 which distributes memory to the various XLH managers 305 for allocating memory to the XLHs.

Detailed Description Text (329):
5. It includes a CRC for an entire "packet" (and another for the header.)

Detailed Description Text (340):
The last word in any packet is a cyclic redundancy check (CRC) word. This word is used to insure the that any corruption of the data in a packet can be detected. The CRC word is computed on all of the data in the packet, excluding any special words like "DLE" that may need to be inserted in the data stream for transparency or other reasons. The polynomial that is used to compute the CRC word is the CRC-16 standard.

Detailed Description Text (371):
Incoming network transaction units are received from the UIMs at their EUSL interface 400 receivers 402, are converted to words in serial to parallel converters 404 and are accumulated in FIFO buffers 94. Each EUSL interface is connected to the NIM transmit bus 95, which consists of a parallel data path, and various signals for bus arbitration and clocking. When a network transaction unit has been buffered, the EUSL interface 400 arbitrates for access to the transmit bus 95. Arbitration proceeds in parallel with data transmission on the bus. When the current data transmission is complete, the bus arbiter awards bus ownership to one of the competing EUSL interfaces, which begins transmission. For each transaction, the EUSL port number, inserted at the beginning of each packet by port number

generator 403, is transmitted first, followed by the network transaction unit. Within an XL interface 440, the XL transmitter 96 provides the bus clock, and performs parallel to serial conversion 442 and data transmission on the upstream XL 3.

Detailed Description Text (383):
EUS Link interfacing. The interface to the EUS Link includes an optical transmitter and receiver, along with the hardware necessary to perform the link level functions required by the EUSL (e.g. CRC generation and checking, data formatting, etc.).

Detailed Description Text (421):
FIG. 19 is a functional overview of the operation and interfaces among the NIM, UIM, and EUS. The specific EUS shown in this illustrative example is a Sun-3 workstation, but the principles apply to other end user systems having greater or lesser sophistication. Consider first the direction from the MINT via the NIM and UIM to the EUS. As shown in FIG. 4, data that is received from MINT 11 over link 3 is distributed to one of a plurality of UIMs 13 over links 14 and is stored in receive buffer memory 90 of such a UIM, from which data is transmitted in a pipelined fashion over an EUS bus 92 having a DMA interface to the appropriate EUS. The control structure for accomplishing this transfer of data is shown in FIG. 19, which shows that the input from the MINT is controlled by a MINT to NIM link handler 520, which transmits its output under the control of router 522 to one of a plurality of NIM to UIM link handlers (N/U LH) 524. MINT/NIM link handler (M/N LH) 520 supports a variant on the Metrobus physical layer protocol. The NIM to UIM link handler 524 also supports the Metrobus physical layer protocol in this implementation, but other protocols could be supported as well. It is possible that different protocols could coexist on the same NIM. The output of the N/U LH 524 is sent over a link 14 to a UIM 13, where it is buffered in receive buffer memory 90 by NIM/UIM link handler 552. The buffer address is supplied by memory manager 550, which manages free and allocated packet buffer lists. The status of the packet reception is obtained by N/U LH 552, which computes and verifies the checksum over header an data, and outputs the status information to receive packet handler 556, which pairs the status with the buffer address received from memory manager 550 and queues the information on a received packet list. Information about received packets is then transferred to receive queue manager 558, which assembles packet information into queues per LUWU and SUWU, and which also keeps a queue of LUWUs and SUWUs about which the EUS has not yet been notified. Receive queue manager 558 is polled for information about LUWUs and SUWUs by the EUS via the EUS/UIM link handler (E/U LH) 540, and responds with notification messages via UIM/EUS link handler (U/E LH) 562. Messages which notify the EUS of the reception of a SUWU also contain the data for the SUWU, thus completing the reception process. In the case of a LUWU, however, the EUS allocates its memory for reception, and issues a receive request via E/U LH 540 to receive request handler 560, which formulates a receive worklist and sends it to resource manager 554, which controls the hardware and effects the data transfer over EUS bus 92 (FIG. 4) via a DMA arrangement. Note that the receive request from the EUS need not be for the entire amount of data in the LUWU; indeed, all of the data may not have even arrived at the UIM when the EUS makes its first receive request. When subsequent data for this LUWU arrives, the EUS will again be notified and will have an opportunity to make additional receive requests. In this fashion, the reception of the data is pipelined as much as possible in order to reduce latency. Following data transfer receive request handler 560 informs the EUS via U/E LH 562, and directs memory manager 550 to de-allocate the memory for that portion of the LUWU that was delivered, thus making that memory available for new incoming data.

Detailed Description Text (467):
FIG. 20 shows the UIM to MINT Message format. The MAN header 610 consists of the Destination Address 612, the Source Address 614, the group (virtual network) identifier 616, group name 618, the type of service 620, the Packet Length (the header plus data in bytes) 622, a type of service indicator 623, a protocol

identifier 624 for use by end user systems for identifying the contents of EUS to EUS header 630, and the Header Check Sequence 626. The header is of fixed length, seven 32-bit words or 224 bits long. The MAN header is followed by an EUS to EUS header 630 to process message fragmentation. This header includes a LUWU identifier 632, a LUWU length indicator 634, the packet sequence number 636, the protocol identifier 638 for identifying the contents of the internal EUS protocol which is the header of user data 640, and the number 639 of the initial byte of data of this packet within the total LUWU of information. Finally, user data 640 may be preceded for appropriate user protocols by the identity of the destination port 642 and source port 644. The fields are 32 bits because that is the most efficient length (integers) for present network control processors. Error checking is performed on the header in control software; this is the Header Check Sequence. At the link level, error checking done over the whole message; this is the Message Check Sequence 634. The NIM/MINT header 600 (explained below) is also shown in the figure for completeness.

Detailed Description Text (472):
9.3.2.2.2 Packet Length

Detailed Description Text (473):
The Packet Length 622 is 16 bits long and represents the length in bytes of this message fragment including the fixed length header and the data. This length is used by the MINT for transmitting the message. It is also used by the destination UIM to determine the amount of data available for delivery to the EUS.

Detailed Description Text (476):
9.3.2.2.4 Packet Sequence Number

Detailed Description Text (477):
This is a Packet Sequence Number 636 for this particular LUWU transmission. It helps the receiving UIM recombine the incoming LUWU, so that it can determine if any fragments of the transmission have been lost because of error. The sequence number is incremented for each fragment of the LUWU. The last sequence number is negative to indicate the last packet of a LUWU. (An SUWU would have -1 as the sequence number.) If an infinite length LUWU is being sent, the Packet Sequence Number should wrap around. (See UWU length, Section 9.3.2.2.7, for an explanation of an infinite length LUWU.)

Detailed Description Text (481):
The UWU ID 632 is a 32 bit number that is used by the destination UIM to recombine a UWU. Note that the recombination job is made easier because fragments cannot get out of order in the network. The UWU ID, along with the Source and Destination Addresses, identifies packets of the same LUWU, or in other words, fragments of the original datagram transaction. The ID must be unique for the source and destination pair for the time that any fragment is in the network.

Detailed Description Text (484):
A length that is negative indicates an infinite length LUWU, which is like an open channel between two EUSs. Closing down an infinite length LUWU is done by sending a negative Packet Sequence Number. An infinite length LUWU only makes sense where the UIM controls the DMA into EUS memory.

Detailed Description Paragraph Table (2):
APPENDIX A ACRONYMS AND ABBREVIATIONS

1SC First Stage Controller 2SC Second Stage Controller ACK Acknowledge ARP Address Resolution Protocol ARQ Automatic Repeat Request BNAK Busy Negative Acknowledge CC Central Control CNAK Control Negative Acknowledge CNet Control Network CRC Cyclic Redundancy Check or Code DNet Data Network DRAM Dynamic Random Access Memory DVMA Direct Virtual Memory Access EUS End User System EUSL End User Link (Connects NIM and UIM) FEP Front End Processor FIFO

First In First Out FNAK Fabric Blocking Negative Acknowledge IL Internal Link (Connects MINT and MANS) ILH Internal Link Handler IP Internet Protocol LAN Local Area Network LUWU Long User Work Unit MAN Exemplary Metropolitan Area Network MANS MAN Switch MANSC MAN/Switch Controller MINT Memory and Interface Module MMU Memory Management Unit NAK Negative Acknowledge NIM Network Interface Module OA&M Operation, Administration and Maintenance PASC Phase Alignment and Scramble Circuit SCC Switch Control Complex SUWU Short User Work Unit TCP Transmission Control Protocol TSA Time Slot Assigner UDP User Datagram Protocol UIM User Interface Module UWU User Work Unit VLSI Very Large Scale Integration VME .RTM. bus An IEEE Standard Bus WAN Wide Area Network XL External Link (Connects NIM to MINT) XLH External Link Handler XPC Crosspoint Controller

---

CLAIMS:

4. The protocol of claim 3 further comprising a user protocol specified by an end user header, comprising:

an identification of a user work unit;

an indication of the length of said user work unit;

an indication of a packet sequence number within said user work unit;

an identification of protocol to be used by said destination user; and

a number of a first byte of said packet within said user work unit;

wherein said destination user system comprises means for identifying said packet with reference to other packets of a user work unit, using said user work unit identification; means for recognizing out of sequence packets using said packet sequence number; said for storing data of said packet relative to a stored address for storing said user work unit using said number of said first byte; and means for recognizing completion of reception of a user work unit using said length indication.

L10: Entry 6 of 10 File: USPT Mar 8, 1994

DOCUMENT-IDENTIFIER: US 5293379 A
** See image for Certificate of Correction **
TITLE: Packet-based data compression method

Application Filing Date (1):
19930527

Detailed Description Text (7):
In FIG. 4, the fields present in a TCP/IP packet are shown, with the number of bytes in each indicated to the left of the FIG. The information contained within each of the fields is shown within each field's boundary. Certain data within the packet remains constant over a multi-packet communication interval. Such information is hereinafter referred to as "static" information and generally remains unchanged for the duration of a conversation or session. While certain header information is altered during a packet's passage through a network, that information is constant for every packet occurring during a conversation. Such information is contained in the time-to-live (TTL) field, where it is decremented as a packet passes through nodes in a network. However, since a LANBRIDGE is stationary, succeeding packets in a conversation have identical TTL fields.

Detailed Description Text (8):
In FIG. 4, static fields comprise header information including a destination address (6 bytes), source address (6 bytes), and packet type (2 bytes). In addition, within the IP header, the internet header length (IHL), type of service, flags, fragment offset, time to live (TTL), protocol, source address and destination address fields are also static. In the TCP header portion of the packet, static fields comprise the source port designation, destination port designation, data offset, flags, window, and urgent pointer.

Detailed Description Text (9):
As aforestated, all of the above data fields tend to remain unchanged over a plurality of packets. While not shown in FIG. 4, it is understood by those skilled in the art that the TCP/IP packet format will generally be received as portion of a higher level packet format which will include further destination and source addresses as well as control data and protocol identification fields. Each of those can also be considered as static data.

Detailed Description Text (10):
A second group of fields within the TCP/IP packet format are termed "recalculatable". In essence, such fields carry information that can be derived from other fields and they are shown in FIG. 4 crossed hatched from upper right to lower left. They include the length, checksum, and frame check sequence (FCS) fields in the IP header.

Detailed Description Text (12):
As regards the length field (number of bytes in the packet), it may be zeroed under certain circumstances, with zeros written-in as placeholders. The length value is regenerated at the receiving end. The checksum value (sum of values in IP header fields) is recalculated by subtracting it from a packet checksum value actually derived from examination of the packet header field values. The result is typically

zero which compresses as a static field. At the receiving end, it is restored by the opposite calculation. By performing the subtraction, rather than just zeroing the checksum field, an error in the original checksum is duplicated at the remote end and lanbridge transparency is retained. A non-zero checksum may occur when the packet header has been computed, when an alternate checksum algorithm is experienced, etc. The checksum field in the TCP header is treated similarly.

Detailed Description Text (28):
The packet's user-data bit length in the output buffer is then compared with the un-reformatted packet user-data bit length still in the input buffer (decision box 68). If the original packet length is shorter, the original input user-data bits are transmitted (box 70) followed by an indicator bit that compression has not occurred (box 71). Otherwise, the output buffer user data bits are transmitted (box 72) and a bit is appended indicating whether the user-data field has been compressed (box 74).

Detailed Description Text (30):
Referring first to FIG. 10, a LAT packet is shown comprising 64 bytes of 8 bit data. The header's first 14 bytes include a destination address, a source address and a type field, and is the same for all Ethernet packets. The "type" field designates a LAT packet in hexidecimal. The next 8 bytes (i.e. "LAT type" through "MsgAck") constitute the remainder of the header.

Detailed Description Text (34):
As shown in FIGS. 13 and 14, a connection table is established and is allocated a row for each conversation in progress. For each conversation (see box 100), the allocated row includes the following values from the last packet transmitted or received in the conversation: the message sequence number; destination sequence number; message acknowledgement number; and source sequence number. Additionally, cross reference tables are established (box 102) which provide LZW dictionary-to-connection pointers and connection table-to-LZW dictionary pointers.

CLAIMS:

21. The method of claim 15, wherein said packet contains a checksum field, said method including the further step of:

c1. calculating a new checksum for said packet, subtracting said new checksum value from the value in said checksum field to obtain a difference value and replacing the value in said checksum field with said difference value.

L10: Entry 5 of 10 File: USPT Jun 27, 1995

DOCUMENT-IDENTIFIER: US 5428470 A
TITLE: Modular system and method for an automatic analyzer

Application Filing Date (1):
19920717

Brief Summary Text (14):
A method for identifying one or modules in an automated system in accordance with the present invention may be used where one or modules are included in a serial communications loop. The method may include the steps of transmitting a message from a master module, the message including a module identification number, receiving the message by a first module after the master module in the serial communications loop, verifying that a module identification number stored in the first module indicates that the first module is not the master module, storing the module identification number, transmitting a message with a module identification number incremented by a predetermined amount, and repeating the receiving step for additional modules in the serial communications loop.

Detailed Description Text (58):
Various communication protocols and logical approaches may be used in implementing a communications loop for the modular system of the present invention. Preferably, all communications on such a communications loop, for example, loop 78 of FIG. 1, is via packets of data. For example, a packet may include a destination address, a source address, a message length, the message itself which may be data, commands or the like, and a checksum value. To indicate that a packet has been successfully received, the receiving module, such as the reagent pump A module 70, transmit an acknowledge packet which has a message length of zero, thus creating a relatively short and therefore quickly transmitted acknowledgment.

Detailed Description Text (65):
In the wait-for-message state 724, the DSP 322 first enters a wait for complete message state 730. When a complete message is received, the DSP 322 checks to determine if the message is a freeze processor count message (step 732). If so, the initialization or reset process of FIG. 6 is completed. The module 300 exits the initialization process and enters a monitor process 734.

Detailed Description Text (66):
If the message is not a freeze processor count message, the DSP 322 checks to determine if the message is a count-off message and if the destination number in the count-off message is greater than the current processor counter variable stored by the DSP 322. If so, the DSP 322 exits the wait-for-message state 724 and enters a count-off state 736.

Detailed Description Text (67):
If the message was not a count-off message with a destination number greater than the current processor counter variable, then a keep alive message has been received (step 738) and the DSP 322 exits the wait-for-message state 724 and enters a send help message state 740.

Detailed Description Text (76):

Similarly, a module further along the serial communications loop 78 within the modules 68 receives count-off messages (step 735), sets up a count-off message (step 736) and begins to send the message packet, only to be interrupted by the beginning of message packets (step 748) from prior modules in the communications loop 78. Each time this occurs the processor counter variable is incremented (step 754), returning the module to the wait-for-message state 724. Ultimately, the module is not interrupted by the receipt of a start-of-packet character while a count-off message is being sent (step 748), thus allowing the complete count-off message to be transmitted and returning the module to the wait-for-message state 724 where the module waits while the remaining modules in the communications loop 78 complete the count-off procedure.

Detailed Description Text (78):
Once the multiple module controller 56 has received count-off messages from all of the modules 68, the controller 56 checks to determine if the number of modules 68 for which the module controller 56 has received count-off messages equals the number of modules that should be in the loop 78 according to system configuration information provided, for example, from the instrument control computer 50. If so, the multiple module controller 56 sends a freeze processor counter variable message to each of the modules 68, specifying each module by processor counter variable value. When this message is received by a module in the wait-for-message state 724, the DSP 322 freezes the processor counter variable, using this value as the module destination address for further communications with the multiple module controller 56, exits the reset process of FIG. 6 at step 732, and enters the module monitor process 734.

Detailed Description Text (80):
The DSP 322, finding that the RAM contents have been initialized (step 704), enters a count resets state 760, incrementing the reset counter variable (step 762), checking the number of resets and, if less than four (step 764), entering the wait-for-message state 724. When the next complete message is received (step 730), the message will not be a freeze processor counter message nor will it be a count-off message (steps 732 and 735) because the modules 68 are not in a count-off procedure. Instead the message is considered a keep-alive message (step 738), that is, any message on the loop 78 from the controller 56 that is not addressed to the particular module, and the DSP 322 sends a help message to the host (step 740). The DSP 322 then returns to the wait-for-message state 724, awaiting the next message from the multiple module controller 56. The controller 56 may respond to the help message as described above.

Detailed Description Text (88):
With the count-off message received from the reagent crane A module 74, the processor in reagent crane B module 76 detects a complete message (step 730) and that the message was a count-off message (step 735). Accordingly, the processor then executes state 736 to form a count-off message and sends the count-off message in state 746 but identifying the processor as number two.

L10: Entry 3 of 10 File: USPT Apr 6, 1999

DOCUMENT-IDENTIFIER: US 5892924 A
TITLE: Method and apparatus for dynamically shifting between routing and switching packets in a transmission network

Application Filing Date (1):
19960131

Detailed Description Text (42):
Separately available ATM components may be assembled into a typical ATM switch architecture. For example, FIG. 5 is a general block diagram of an architecture of an ATM switch 3 (the example shows a 16-port switch) that may be used as the switching hardware engine of a basic switching unit according to an embodiment of the present invention. However, commercially available ATM switches also may operate as the switching engine of the basic switching unit according to other embodiments of the present invention. The main functional components of switching hardware 3 include a switch core, a microcontroller complex, and a transceiver subassembly. Generally, the switch core performs the layer 2 switching, the microcontroller complex provides the system control for the ATM switch, and the transceiver subassembly provides for the interface and basic transmission and reception of signals from the physical layer. In the present example, the switch core is based on the MMC Networks ATMS 2000 ATM Switch Chip Set which includes White chip 100, Grey chip 102, MBUF chips 104, Port Interface Device (PIF) chips 106, and common data memory 108. The switch core also may optionally include VC Activity Detector 110, and Early Packet Discard function 112. Packet counters also are included but not shown. White chip 100 provides configuration control and status. In addition to communicating with White chip 100 for status and control, Grey chip 102 is responsible for direct addressing and data transfer with the switch tables. MBUF chips 104 are responsible for movement of cell traffic between PIF chips 106 and the common data memory 108. Common data memory 108 is used as cell buffering within the switch. PIF chips 106 manage transfer of data between the MBUF chips to and from the switch port hardware. VC Activity Detector 110 which includes a memory element provides information on every active virtual channel. Early Packet Discard 112 provides the ability to discard certain ATM cells as needed. Packet counters provide the switch with the ability to count all packets passing all input and output ports. Buses 114, 115, 116, 117, and 118 provide the interface between the various components of the switch. The microcontroller complex includes a central processing unit (CPU) 130, dynamic random access memory (DRAM) 132, read only memory (ROM) 134, flash memory 136, DRAM controller 138, Dual Universal Asynchronous Receiver-Transmitter (DUART) ports 140 and 142, and external timer 144. CPU 130 acts as the microcontroller. ROM 134 acts as the local boot ROM and includes the entire switch code image, basic low-level operation system functionality, and diagnostics. DRAM 132 provides conventional random access memory functions, and DRAM controller 138 (which may be implemented by a field programmable gate array (FPGA) device or the like) provides refresh control for DRAM 132. Flash memory 136 is accessible by the microcontroller for hardware revision control, serial number identification, and various control codes for manufacturability and tracking. DUART Ports 140 and 142 are provided as interfaces to communications resources for diagnostic, monitoring, and other purposes. External timer 144 interrupts CPU 130 as is required. Transceiver subassembly includes physical interface devices 146, located between PIF chips 106 and physical

transceivers (not shown). Interface devices 146 perform processing of the data stream, and implement the ATM physical layer. Of course, the components of the switch may be on a printed circuit board that may reside on a rack for mounting or for setting on a desktop, depending on the chassis that may be used.

Detailed Description Text (55):
In an embodiment of the present invention, three flow types are specified: Flow Type 0, Flow Type 1, and Flow Type 2. Of course, different or additional flow types also may be specified. Flow Type 0 is used to change the encapsulation of IP packets from the default encapsulation. The format of a flow identifier for Flow Type 0 is null and accordingly has a zero length. Flow Type 1 is a flow type that specifies the set of fields from the packet header that identify the flow as having packets carrying data between applications running on stations. Flow Type 1 is useful for flows having packets for protocols such as UDP and TCP in which the first four octets after the IP header specify a source port number and a destination port number that are used to indicate applications. A flow identifier for Flow Type 1 has a length of four 32-bit words. The format of a flow identifer for Flow Type 1, indicated as reference number 240 shown in FIG. 7a, includes (described in order of most significant bit (MSB) to least significant bit (LSB)) the Version, Internet Header Length (IHL), Type of Service, and Time to Live, and Protocol fields as the first word; the Source Address field as the second word; and the Destination Address field as the third word. These fields in the flow identifier are from the header of the IP packet of Flow Type 1. The flow identifer for Flow Type 1 also includes the Source Port Number and the Destination Port Number fields (the first four octets in the IP packet after the IP header) as the fourth word. Flow Type 2 is a flow type that specifies the set of fields from the packet header that identify the flow as having packets carrying data between stations without specifying the applications running on the stations. A flow identifier for Flow Type 2 has a length of three 32-bit words. The format of a flow identifer for Flow Type 2, indicated by reference number 250 shown in FIG. 7b, includes the Version, Internet Header Length (IHL), Type of Service, Time to Live, Protocol, Source Address, and Destination Address fields from the header of the IP packet. The format of a flow identifier for Flow Type 2 is the same as that for Flow Type 1 without the fourth word. The hierarchical nature of the flow identifiers for the various flow types allows a most specific match operation to be performed on an IP packet to facilitate flow classification.

Detailed Description Text (67):
FIG. 8a illustrates the structure of a generic IFMP adjacency protocol message 300. All IFMP adjacency protocol messages are encapsulated within an IP packet. FIG. 8b illustrates a generic IP packet (in its current version IPv4) with a variable length Data field into which an IFMP adjacency protocol message may be encapsulated. As an indication that the IP packet contains an IFMP message, the Protocol field in the IP header of the encapsulating IP packet must contain the decimal value 101. The Time to Live field in the header of the IP packet encapsulating the IFMP message is set to 1. Also, all IFMP adjacency protocol messages are sent to the limited broadcast IP Destination Address (255.255.255.255), using the address in the Destination Address field of the IP header. As seen in FIG. 8a, an IFMP adjacency protocol message 300 includes (described in order of MSB to LSB) the following fields: an 8-bit Version (302), an 8-bit Op Code (304), and a 16-bit Checksum (306) as the first 32-bit word; Sender Instance (308) as the second 32-bit word; Peer Instance (310) as the third 32-bit word; Peer Identity (312) as the fourth 32-bit word; Peer Next Sequence Number (314) as the fifth 32-bit word; and Address List (316) which is a field of a variable number of 32-bit words.

Detailed Description Text (69):
Checksum 306 is the 16-bit one's complement of the one's complement sum of: the source address, destination address and protocol fields from the IP packet encapsulating the IFMP adjacency protocol message, and the total length of the IFMP

adjacency protocol message. Checksum 306 is used by the system for error control purposes.

Detailed Description Text (80):
FIG. 9a illustrates the structure of a generic IFMP redirection protocol message 380. Like all IFMP adjacency protocol messages, all IFMP redirection protocol messages are encapsulated within an IP packet. FIG. 8b illustrates a generic IP packet (in its current version IPv4) with a variable length Data field into which an IFMP redirection protocol message may be encapsulated. As an indication that the IP packet contains an IFMP message, the Protocol field in the IP header of the encapsulating IP packet must contain the decimal value 101, and the Time to Live field in the header of the IP packet encapsulating the IFMP message is set to 1. An IFMP redirection protocol message is sent to the IP address of the peer at the other end of the link (the IP address being obtained from the IFMP adjacency protocol), using the IP address in the Destination Address field of the IP header. As seen in FIG. 9a, an IFMP redirection protocol message 380 includes (described in order of MSB to LSB) the following fields: an 8-bit Version (382), an 8-bit Op Code (384), and a 16-bit Checksum (386) as the first 32-bit word; Sender Instance (388) as the second 32-bit word; Peer Instance (390) as the third 32-bit word; Sequence Number (392) as the fourth 32-bit word; and a Message Body (394) which is a field of a variable number of 32-bit words.

Detailed Description Text (82):
Checksum 386 is the 16-bit one's complement of the one's complement sum of: the source address, destination address and protocol fields from the IP packet encapsulating the IFMP redirection protocol message, and the total length of the IFMP redirection protocol message. Checksum 386 is used by the system for error control purposes.

Detailed Description Text (101):
In the present invention, the default encapsulation for IP packets on ATM data links is the Logical Link Control/SubNetwork Attachment Point (LLC/SNAP) encapsulation shown in FIG. 10b. FIG. 10b illustrates a default encapsulated IP packet 480. Basically, the default encapsulation prefixes an LLC/SNAP header to the IP packet which is encapsulated within the payload of an ATM Adaptation Layer type 5 Common Part Convergence Sublayer Protocol Data Unit (AAL-5 CPCS-PDU). Described from MSB to LSB, default encapsulated IP packet 480 includes an LLC/SNAP header (24-bit LLC field 482 followed by an 8-bit portion of SNAP header 484 in the first 32-bit word, and the remaining 32-bit word portion of SNAP header 484), IP packet 486 (which has a length of an integer multiple of 32-bit words), Pad field 488, and AAL-5 CPCS-PDU Trailer field 490. Pad field 488 may range from 0 to 47 octets, and Trailer field 490 is 8 octets (four 32-bit words). The MTU of the IP packet 486 using default encapsulation is 1500 octets. The packets using default encapsulation are sent to VPI=0, VCI=1 (forwarded on default virtual channel).

Detailed Description Text (102):
Flow Type 0 encapsulation is used to change the encapsulation of IP packets from the default encapsulation. FIG. 10c illustrates a Flow Type 0 encapsulated IP packet 492. IP packets using Flow Type 0 are encapsulated directly in the payload of an AAL-5 CPCS-PDU without a prefixed LLC/SNAP header. Described from MSB to LSB, Flow Type 0 encapsulated IP packet 492 includes IP packet 494 (which has a length of an integer multiple of 32-bit words), Pad field 496, and AAL-5 CPCS-PDU Trailer field 498. Pad field 496 may range from 0 to 47 octets, and Trailer field 498 is 8 octets (four 32-bit words). The MTU of the IP packet 494 using Flow Type 0 encapsulation is 1500 octets. The packets belonging to the flow redirected from the default virtual channel use Flow Type 0 encapsulation and are sent to the VPI/VCI specified in the Label field of the IFMP REDIRECT message element encapsulated in IP packet 494 (the IFMP REDIRECT message element encapsulated in IP packet 494 is sent in Flow Type 0 encapsulation).

Detailed Description Text (104):
Flow Type 1 is used for packets carrying data between applications running on stations. FIG. 10d illustrates a Flow Type 1 encapsulated IP packet. IP packets using Flow Type 1 encapsulation are essentially disassembled and selected portions of the disassembled IP packet are encapsulated directly in the payload of an AAL-5 CPCS-PDU, without a prefixed LLC/SNAP header. Described from MSB to LSB, Flow Type 1 encapsulated IP packet 500 includes 16-bit Total Length field 502 and 16-bit Identification field 504 from the IP header of the disassembled IP packet, as a first 32-bit word. The value of the Total Length field 502 is not changed, but remains the total length of the IP packet before disassembly. Flow Type 1 encapsulated IP packet 500 also includes the 8-bit Flags field 506, 12-bit Fragment Offset field 508, and 16-bit Checksum field 510 from the IP header of the disassembled IP packet, as a second 32-bit word. The transmitted value of Checksum field 510 is the checksum value that would have been computed for the entire IP header if the TTL field had been set to zero. The Version, IHL, TOS, TTL, Protocol, Source Address, and Destination Address fields in the IP header are not transmitted as part of the Flow Type 1 encapsulated IP packet 500. In addition, the first four octets immediately following the IP header (as determined by the IHL) are not transmitted as part of the Flow Type 1 encapsulated IP packet 500. These first four octets correspond to the source port and destination port for TCP and UDP datagrams, as an example. The source port and destination port fields identify the applications running on the stations. Further, Flow Type 1 encapsulated IP packet 500 includes Data 512. Data field 512 is followed by Pad field 514 and AAL-5 CPCS-PDU Trailer field 516. Pad field 514 may range from 0 to 47 octets, and Trailer field 516 is 8 octets (four 32-bit words). The MTU of the IP packet using Flow Type 1 encapsulation is 1484 octets. The packets belonging to the flow redirected using Flow Type 1 encapsulation are sent to the VPI/VCI specified in the Label field of the corresponding Flow Type 1 IFMP REDIRECT message element encapsulated in the disassembled IP packet (the Label field may be configured to correspond to the source and destination port fields in the TCP or UDP messages).

Detailed Description Text (105):
Flow Type 2 is used for packets carrying data between stations without regard to what applications are running on the stations. FIG. 10e illustrates a Flow Type 2 encapsulated IP packet. IP packets using Flow Type 2 encapsulation are essentially disassembled and selected portions of the disassembled IP packet are encapsulated directly in the payload of an AAL-5 CPCS-PDU, without a prefixed LLC/SNAP header. Described from MSB to LSB, Flow Type 2 encapsulated IP packet 520 includes 16-bit Total Length field 522 and 16-bit Identification field 524 from the IP header of the disassembled IP packet, as a first 32-bit word. The value of the Total Length field 522 is not changed, but remains the total length of the IP packet before disassembly. Flow Type 2 encapsulated IP packet 520 also includes the 8-bit Flags field 526, 12-bit Fragment Offset field 528, and 16-bit Checksum field 530 from the IP header of the disassembled IP packet, as a second 32-bit word. The transmitted value of Checksum field 530 is the checksum value that would have been computed for the entire IP header if the TTL field had been set to zero. The Version, IHL, TOS, TTL, Protocol, Source Address, and Destination Address fields in the IP header are not transmitted as part of the Flow Type 2 encapsulated IP packet 520. Unlike Flow Type 1 encapsulation, the first four octets immediately following the IP header (as determined by the IHL) are transmitted as part of the Flow Type 2 encapsulated IP packet 520. Further, Flow Type 2 encapsulated IP packet 520 includes Data 532. Data field 532 is followed by Pad field 534 and AAL-5 CPCS-PDU Trailer field 536. Pad field 534 may range from 0 to 47 octets, and Trailer field 536 is 8 octets (four 32-bit words). The MTU of the IP packet using Flow Type 2 encapsulation is 1488 octets. The packets belonging to the flow redirected using Flow Type 2 encapsulation are sent to the VPI/VCI specified in the Label field of the corresponding Flow Type 2 IFMP REDIRECT message element encapsulated in the disassembled IP packet.

Detailed Description Text (112):

In the present invention, GSMP packets are variable length and encapsulated directly in an AAL-5 CPCS-PDU with a prefixed LLC/SNAP header, in a similar manner as the default encapsulation for IP packets on ATM data links described above in relation to FIG. 10b. FIG. 11a illustrates an encapsulated GSMP packet 540. Basically, the default encapsulation prefixes an LLC/SNAP header to the GSMP packet which is encapsulated within the payload of an AAL-5 CPCS-PDU. Described from MSB to LSB, default encapsulated GSMP packet 540 includes an LLC/SNAP header (24-bit LLC field 542 followed by an 8-bit portion of SNAP header 544 in the first 32-bit word, and the remaining 32-bit word portion of SNAP header 544), GSMP message 546 (which has a length of an integer multiple of 32-bit words), Pad field 548, and AAL-5 CPCS-PDU Trailer field 550. Pad field 548 may range from 0 to 47 octets, and Trailer field 550 is 8 octets (four 32-bit words). The MTU of the GSMP message 546 using default encapsulation is 1500 octets. The packets using default encapsulation are sent to VPI=0, VCI=1 (default virtual channel).

Detailed Description Text (126):
The Add Branch message is a GSMP CM message used to establish a virtual channel connection or to add an additional branch to an existing virtual channel connection. In present embodiment, no distinction is made between unicast and multicast connections. A first Add Branch message for a particular Input Port, Input VPI, and Input VCI establishes a unicast connection. A second Add Branch message with the same Input Port, Input VPI, and Input VCI converts the unicast connection to a multicast connection by adding another output branch. Other output branches may be added in the same manner with further Add Branch messages. Also, an Add Branch message may be used to check the connection state stored in the ATM switch. The Delete Branch message is a GSMP CM message used to delete a single branch of a virtual channel connection. For example, use of Delete Branch message on a multicast virtual channel connection with two branches removes a branch converting the multicast connection into a unicast connection. The Delete Branch message may also be used to delete a connection by deleting the last branch in a virtual channel connection. Another GSMP CM message, the Delete Tree message is used to delete an entire virtual connection by deleting all remaining branches of the connection. The Verify Tree message is a GSMP CM message used to verify the number of branches on a virtual channel connection. The Delete All message is a GSMP CM message that is used to delete all connections on a switch input port. The Move Root message is a GSMP CM message used to move an entire virtual connection tree from its current Input Port, Input VPI, and Input VCI, to a new Input Port, Input VPI and Input VCI. Another GSMP CM message, the Move Branch message is used to move a single output branch of a virtual channel connection from its current Output Port, Output VPI, and output VCI, to a new Output Port, Output VPI, and Output VCI on the same virtual channel connection.

Detailed Description Text (133):
For GSMP CM messages, Port Session Number field 632 provides the session number of the input port. In particular, the value in Port Session Number field 632 gives the port session number of the switch input port indicated in Input Port field 634. Each switch port maintains a port session number that is assigned by the switch. The port session number remains unchanged while the port is continuously up. However, a new and different port session number is generated after a port is up after being down or unavailable. It is preferred that the new port session number be randomly selected. If the switch controller sends a GSMP CM request message that has an invalid value in Port Session Number field 632, then the switch rejects the GSMP CM request message by sending a GSMP CM failure response message with Code field 628 indicating an invalid port session number causing the failure. A current port session number may be obtained using a GSMP Configuration message.

Detailed Description Text (136):
For a GSMP CM message, Number of Branches field 650 gives the number of output branches on a virtual channel connection. Field 650 is used in a GSMP CM Verify Tree message. For all other GSMP CM messages, field 650 is set to zero by the

sender entity and ignored by the receiver entity. In the present embodiment, Reserved field 652 which is not used for GSMP CM messages is set to zero by the sender entity and ignored by the receiver entity.

Detailed Description Text (141):
The Verify Tree message is a GSMP CM message used to verify the number of branches on a virtual channel connection. The connection is specified by Input Port field 634, Input VPI field 638, and Input VCI field 640. Output Port field 642, Output VPI field 646, and Output VCI field 648 in a Verify Tree message are not used, and are set to zero by the switch controller and ignored by the switch. The number of branches that the switch believes the specified virtual channel connection should contain is given by Number of Branches field 650 in the Verify Tree request message. FIG. 13d is a general diagram illustrating the operation of an ATM switch that receives a GSMP Verify Tree request message from switch controller. At step 730, switch controller sends a GSMP Verify Tree request message that is received by the ATM switch. The ATM switch determines whether the virtual channel connection, as specified in Input Port field 634, Input VPI field 638, and Input VCI field 640 of the received Verify Tree request message, exists in the switch, at a step 732. If the switch determines at step 732 that the virtual channel connection does exist, then the switch at step 734 checks the actual number of branches for the specified connection and compares the actual number with that in Number of Branches field 650 of the received Verify Tree request message. If the switch determines at step 736 that the numbers match then the verification operation was successful. If the verification is determined to be successfully completed, the switch at step 738 determines from Result field 626 of the Verify Tree request message whether a response is required when the request is successful. If the Result field of the request message indicates AckAll (success response required), the switch sends a Verify Tree success response message to the switch controller in step 740. The Verify Tree success response message is a copy of the received Verify Tree request message with Result field 626 indicating Success. If it is determined at step 738 that a success response is not required, then the switch has no response (indicated as 742). If the switch determines at step 732 that the connection specified in the Verify Tree request message does not exist, then the switch at step 744 sends a Verify Tree failure response message to the switch controller with the appropriate failure code. A Verify Tree failure response message is a copy of the received Verify Tree request message with Result field 626 indicating Failure and with the type of failure indicated by the appropriate failure code in its Code field 628. If the switch determines at step 736 that the verification operation is unsuccessful, then the switch at step 746 sets the actual number of branches into the Number of Branches field 650 of the Verify Tree failure response message and sends it to the switch controller with the Code field 628 set to zero.

Detailed Description Text (145):
Providing switch port management, the GSMP Port Management (PM) message allows a port to be brought into service, taken out of service, looped back, or reset. FIG. 14 illustrates the structure for a GSMP PM message 870, used as both request and response messages. GSMP PM message 870 may be contained in GSMP Message field 546 of the encapsulated GSMP packet 540 in FIG. 11a. As seen in FIG. 14, GSMP PM message 870 includes (described in order of MSB to LSB) the following fields: an 8-bit Version field 622, an 8-bit Message Type field 624, an 8-bit Result field 626, and an 8-bit Code field 628; 32-bit word Transaction Identifier field 630; 32-bit word Port field 872; 32-bit word Port Session Number field 874; 32-bit Event Sequence Number field 874; 8-bit Events Flag field 878; 8-bit Duration field 880; and 16-bit Function field 882. Version field 622, Message Type field 624, Result field 626, Code field 628, Transaction Identifier field 630, and Port Session Number 874 are used generally in the same manner as for other GSMP messages, as discussed earlier. Port field 872 gives the port number of the port to which the GSMP PM message applies. A GSMP PM message has a particular Message Type field and various possible functions that may be specified in Function field 882. Some of the functions of GSMP PM messages include: a Bring Up function, a Take Down function,

an Internal Loopback function, an External Loopback function, a Bothway Loopback function, a Reset Input Port function, and a Reset Event Flags function. Each switch port maintains an Event Sequence Number and a set of Event Flags (one Event Flag for each type of Event Message). The Event Sequence Number is set to zero when the port is initialized and is incremented each time an asynchronous event reportable by an Event message is detected on that port, regardless of whether the Event message is sent or not. When a switch port sends an Event message, it sets the corresponding Event Flag on that port. The port is not permitted to send another Event message of the same type until the corresponding Event Flag is reset by a Reset Event Flags function of a GSMP PM message. The use of the Event Flags provides simple flow control to prevent the switch from flooding the switch controller with Event messages. In a GSMP PM request message, Event Sequence Number field 876 is not used and is set to zero by the switch controller and ignored by the switch. In a GSMP PM success response message, Event Sequence Number field 876 gives the current value of the Event Sequence Number of the switch port specified in the received GSMP PM request message. In a GSMP PM request message with the Function field 882 specifying Reset Event Flags, particular bits in the Event Flags field 878 may be used to reset the corresponding Event Flags in the switch port specified by the Port field 872. In a GSMP PM success response message with the Function field 882 specifying Reset Event Flags, the bits in Event Flags field 878 are set to the current values of the corresponding Event Flags for the specified port, after the Event Flags specified in the request message have been reset. By setting the Event Flags field to all zeros in a GSMP PM message with a Reset Event Flags function, the switch controller is able to obtain the current state of the Event Flags and the current Event Sequence Number of the specified port without changing the state of the Event Flags. In other GSMP PM messages with a different Function field 882 specified, the Event Flags field 878 is not used and is set to zero by the switch controller and ignored by the switch. Duration field 880 is used only in GSMP PM messages with the Function field 882 specified as Internal Loopback, External Loopback, or Bothway Loopback. Duration field 880 provides the length of time (in seconds) that any of the loopback states remains in operation. When the duration expires, the port which was in loopback automatically returns to service. In GSMP PM messages with a different Function field 882 specified, Duration field 880 is not used and is set to zero by the switch controller and ignored by the switch. In GSMP PM messages, Function field 882 specifies the action to be taken (the specified action is taken regardless of the current status of the port). The Bring Up function brings the port into service, and the Take Down function takes the port out of service. The Internal Loopback function performs an internal loopback (ATM cells arriving at the output port from the switch fabric are looped through to the input port back to the switch fabric). The External Loopback function performs an external loopback (ATM cells arriving at the input port from the external communications link are looped back to the communications link at the physical layer without entering the input port). The Bothway Loopback function performs both internal and external loopback. The Reset Input Port function resets the input port (all connections arriving at the specified input port are deleted and the input and output port hardware are reinitialized so that all VPI/VCI values for the specified input port in the connection table are empty). The Reset Event Flags function resets the Event Flags as discussed above.

Detailed Description Text (147):
GSMP Configuration messages permit the switch controller to determine the capabilities of the ATM switch in basic switching unit. Three message types for GSMP Configuration messages are defined: Switch Configuration, Port Configuration, and All Ports Configuration. GSMP Configuration messages use different formats for the request message and the response message, since they contain different information in their fields. Sent by switch controller to an ATM switch, a Switch Configuration request message, indicated by a particular Message Type field, asks the ATM switch for its global configuration. Then the switch returns to the switch controller a Switch Configuration response message that includes fields for the switch type and switch name of the ATM switch, as well as the version of the switch

control firmware installed. The switch type is a allocated by a manufacturer of the switch to identify the switch product, and the switch name may be a 48-bit IEEE 802 MAC address or other quantity that is unique within the operational context of the switch. A Port Configuration request message has its own particular Message Type field and is sent by switch controller to an ATM switch. The Port Configuration request message asks the switch for configuration information of a single switch port that is specified in the Port field of a Port Configuration request message. The switch sends to the switch controller a Port Configuration success response message that includes configuration information for both the input and output sides of the specified port. The configuration information in a Port Configuration success response message includes: the current Port Session Number of the port, the minimum value of VPI that the connection table on the input port that can be supported by GSMP, the maximum value of VPI that the connection table on the input port that can be supported by GSMP, the minimum value of VCI that the connection table on the input port that can be supported by GSMP, and the maximum value of VCI that the connection table on the input port that can be supported by GSMP. The configuration information also includes: the cell rate (rate of ATM cells per second) of the port, the current status (i.e., down, up, unavailable, internal loopback, external loopback, or bothway loopback) of the port; the port type (the type of physical transmission interface of the port, e.g., unknown, SONET STS-3c at 155.52 Mbps, DS3 at 44.736 Mbps, 4B/5B encoding at 100 Mbps, 8B/10B encoding at 155.52 Mbps, 25 Mbps ATM Forum physical layer, or 51 Mbps ATM Forum physical layer); and the number of priorities that the output port can assign to virtual channel connections. The configuration information provided is referred to as the Port Record for a port. The switch controller sends an All Ports Configuration request message, which has its own particular Message Type field, to the ATM switch to ask for the configuration information for all of the switch ports. Thus, the All Ports Configuration request message does not specify a particular port. The switch sends an All Ports Configuration success response message that provides: the number of Port Records contained by the response message, the byte length of each Port Record, and the Port Records for each port. The Port Record for each port is the same configuration information discussed for the Port Configuration success response message. Of course, if the number of Port Records exceeds a specified maximum amount set for the All Ports Configuration success response message, then the Port Records may be sent in multiple success response messages that each do not exceed the specified maximum amount.

Detailed Description Text (148):
GSMP Event messages allow the ATM switch to inform the switch controller of certain asynchronous events. As mentioned earlier, Event messages are not acknowledged. Event messages may have different Message Types, depending on the asynchronous event. Different Event messages include a Port Up Event message, a Port Down Event message, an Invalid VPI/VCI Event message, a New Port Event message, and a Dead Port Event message. Each switch port maintains an Event Sequence Number and a set of Event Flags (one Event Flag for each type of Event Message). When a switch port sends an Event message, it sets the corresponding Event Flag on that port. The port is not permitted to send another Event message of the same type until the corresponding Event Flag is reset by a Reset Event Flags function of a GSMP Port Management message. The use of the Event Flags provides simple flow control to prevent the switch from flooding the switch controller with Event messages. The Event Sequence Number is set to zero when the port is initialized and is incremented each time an asynchronous event reportable by an Event message is detected on that port, regardless of whether the Event message is sent or not. The current Event Sequence Number is included in Event messages to inform the switch controller of asynchronous events that have occurred on the port, but that have not been reported via an Event message due to the action of the simple flow control mechanism. A Port Up Event message informs the switch controller that the specified port has changed from the down state to the up state. When a port comes up, all connections on its input port are deleted (the input port's connection tables are empty) and a new Port Session Number is assigned by the switch. A Port Down Event

message informs the switch controller that the specified port has changed from the up state to the down state. If a switch is capable of detecting link failure, the switch sends a Port Down Event message to report link failure to the switch controller. When one or more ATM cells arrive at an input port with a VPI/VCI that is not currently allocated to a virtual channel connection, the switch sends an Invalid VPI/VCI Event message to the switch controller. The Invalid VPI/VCI Event message specifies the input port and the VPI/VCI in the Port and VPI/VCI fields respectively. A New Port Event message specifying the number of a new port informs the switch controller that the new port has been added to the switch. The Dead Port Event message informs the switch controller that a port has been removed from the switch. The Dead Port Event message specifies the number of the removed port and the Port Session Number that was valid before the port was removed in its Port and Port Session Number fields respectively.

Detailed Description Paragraph Table (2):
TABLE 2 Switch Components
SWITCH CORE Core chip set MMC Networks ATMS 2000 ATM Switch Chip Set (White chip, Grey chip, MBUF chips, PIF chips) Common data memory standard memory modules Packet counters standard counters MICROCONTROLLER COMPLEX CPU Intel 960CA/CF/HX DRAM standard DRAM modules ROM standard ROM Flash memory standard flash memory DRAM controller standard FPGA, ASIC, etc. DUART 16552 DUART External timer standard timer TRANSCEIVER SUBASSEMBLY Physical interface PMC-Sierra PM5346